Code

Calculating IBS & IBD

Nick Brazeau, Sophie Berube, Izzy Routledge, & Abebe Fola

August 05, 2022

Dependencies for Practical

Please copy and paste the below code chunk in it's entirety to your console to download R package libraries needed for this practical. If you are having trouble installing any of the R packages, please ask an instructor for a pre-loaded flash drive.

Hide

```
deps <- c("tidyverse", "vcfR", "MIPanalyzer", "hmmibdr", "sf", "cowplot", "tidygraph", "ggraph")
deps <- !sapply(deps, function(x){x %in% installed.packages()[,1]} )
if(any(deps)) {
  if(deps["hmmibdr"]) {
    if (!"remotes" %in% installed.packages()[,1]){
      install.packages("remotes")
    }
    remotes::install_github("OJWatson/hmmibdr")
    deps <- deps[names(deps) != "hmmibdr"]
  } else if(deps["MIPanalyzer"]) {
    if (!"remotes" %in% installed.packages()[,1]){
      install.packages("remotes")
    }
    remotes::install_github("mrc-ide/MIPanalyzer")
    deps <- deps[names(deps) != "MIPanalyzer"]
  } else {
    install.packages(names(deps)[deps])
  }
}
```

Please now load all of those libraries into this session using the code chunk below. Please copy and paste it in its entirety.

Hide

```
library(tidyverse)
library(vcfR)
library(hmmibdr)
library(sf)
library(tidygraph)
library(ggraph)
library(cowplot)
```

Finally, please source (*i.e.* load) the file called `utils.r` that is stored under the `IBD/R` directory. If you are using the `IBD.Rproj` environment, you can just run `source("R/utils.R")` as below. Or if you are running from a different environment or working directory (`getwd()`), use the `file.choose` function to help locate the file.

Hide

```
source("R/utils.R")
```

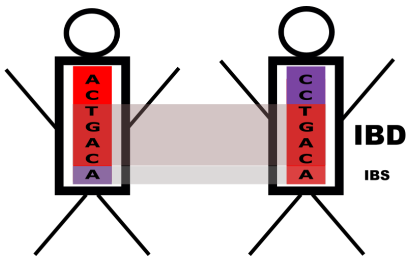
Introduction

Relatedness Overview

During prior sessions, you have explored how to estimate relationships based on levels of population diversity, structure, and connectivity. In this practical, we will focus on between-parasite, or pairwise, measures of genetic relatedness. These are more "individual" focused measures of relatedness versus population, or deme, focused measures of relatedness.
There are two mains ways to estimate relatedness among individuals: identity by descent (IBD) or Time to Most Recent Common Ancestor (TMRCA) (see Speed & Balding 2015 (https://pubmed.ncbi.nlm.nih.gov/25404112/) for further discussion). We will be focusing on IBD, which leverages recombination to detect recent common inheritance of genetic material, or genetic relatedness.

IBS/IBD Overview

As described in the lecture, we can consider pairwise relatedness by determining if the genetic sequence at a give position in the genome, a loci, between two parasites is identical (e.g. the same allele). We can either simply measure the number of sites with identical alleles between two individuals, termed identity by state (IBS), or, we can use statistical models to determine if identical alleles and "blocks" of the genome were likely to be inherited from a common ancestor, termed identity by descent (IBD). It is important to note the differences between IBS and IBD: IBS refers to the "state" or realization of the allele at a loci and can be the result of numerous factors other than ancestry (e.g. selection, drift, structure, chance). In contrast, IBD solely refers to inherited genetic material, which follow specific patterns relevant to public health that we will explore below. The differences between IBS and IBD are visualized in the schematic below, where sites may be IBS but not IBD, whilst all sites that are IBD are IBS.



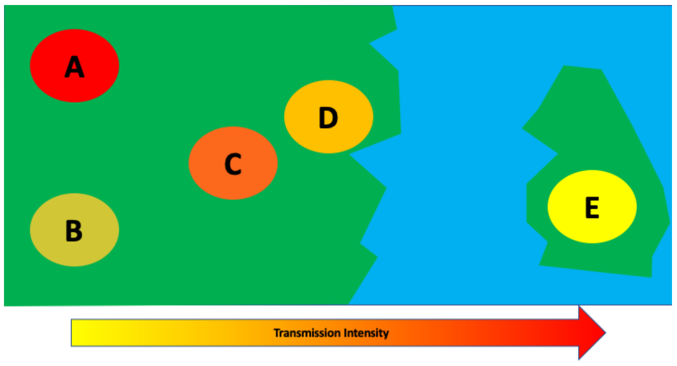
Overview of Data

This data was simulated using a simple malaria model that takes into account spatial relationships, complexity of infection (COI), and population size under a Wright-Fisher model: PolySimIBD (https://github.com/nickbrazeau/polySimIBD) - originally published in Verity/Aydemir/Brazeau et al. 2020 (https://pubmed.ncbi.nlm.nih.gov/32355199/). *Note*, the simulated data has a single chromosome that is only 1,000 base pairs long (much smaller than malaria chromosomes) and as a result, has a much smaller recombination rate (relevant for `hmmIBD` input parameters.)
For this practical, we have simulated five populations, or demes: A-E. The demes were simulated to have the same number of people, or hosts; however, we have randomly selected five individuals from each deme to undergo molecular surveillance and be "sequenced" (*i.e.* in real life, you may select five participants from a village to donate whole-blood vs collecting blood from every individual in the village). The demes have varying intensity of transmission, or incidence, as indicated by the spectrum of yellow-red in the schematic legend. In addition, we expect for mosquitoes to migrate between the demes with respect to how far the demes are from each other (*i.e.* isolation by distance). Finally, we *hypothesize* that the ocean between demes A-D and E is a barrier to any transmission from the "mainland" to the "island" (*i.e.* we expect mainland parasites to be distinct from island parasites). In summary, the data consists of:

- Five demes (4 mainland, 1 island)
- Locations are known for each deme
- Five random samples from each deme

- Incidence as estimated by mean COI per deme*

**As discussed yesterday, COI is a poor measure of incidence in the real world. For our model, it is quite correlated (in expectation) and will be used as a proxy here.*



Practical Goals

By the end of this practical, you should able to perform the below tasks and understand the following concepts:
+ Calculate IBS
+ Calculate IBD
+ Understand strengths and weaknesses of IBS + Understand strengths and weaknesses of IBD

---

1 Importing Data

Here, we will use the `vcfR` package (https://knausb.github.io/vcfR_documentation/) to read in our variant call file (VCF) that contains fifty biallelic SNPs for our twenty-five individuals across our five populations (A, B, C, D, E; *individuals named* A1-5, B1-5, …).

Using the skills that you acquired yesterday, read in the VCF from our data folder that you downloaded from Github: `data/simulated_ibd.vcf.gz` . If you are having trouble locating the file on your computer, please type `file.choose` into your console for an interactive option to locate the file. Name the object (the vcf file you are reading in) “vcf” for simplicity.

Hide

```
# participants DO NOT need to include the quiet bit
vcf <- quiet(vcfR::read.vcfR("data/simulated_ibd.vcf.gz"))
```

1.1 Initial QC

As is always good practice, we can check our VCF for missing data and confirm it has the number of SNPs and samples that we expect. Explore the details and inner-workings of our new VCF to get a sense if it is trustworthy (*i.e.* does the data look reasonable? Is there anything strange in the INFO column?).

Coding Question 1: *Use the* `extract.gt` *function from the vcfR package to pull out the allele depth for the first record (referent allele) for your new VCF. Now plot that allele depth as an initial data exploration using the* `heatmap.bp` *function from the* `vcfR` *package to generate the figure below.*

Click For Answer

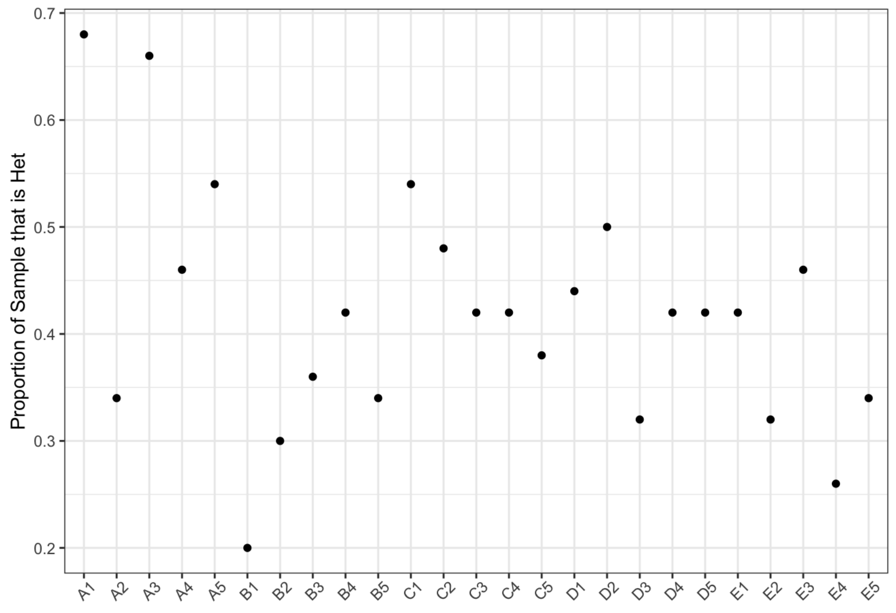
Conceptual Question 1: *Is above plot consistent w/ a high quality VCF? What are some questionable features?*

Click For Answer

1.2 Exploring Hets

Here, we will explore the number of heterzygous genotype calls we have per sample in our VCF. Remember, heterzygous genotype calls give us a crude approximation of COI.

Coding Question 2: Again, use the `extract.gt` function from the vcfR package and pull out the genotype calls (`element = "GT"`) for all samples. Then wrangle the data into “long” format using `pivot` longer. Once you have “tidy long” data, calculate the mean heterozygosity per sample using the `summarise` function and plot the data in a scatterplot, as below.



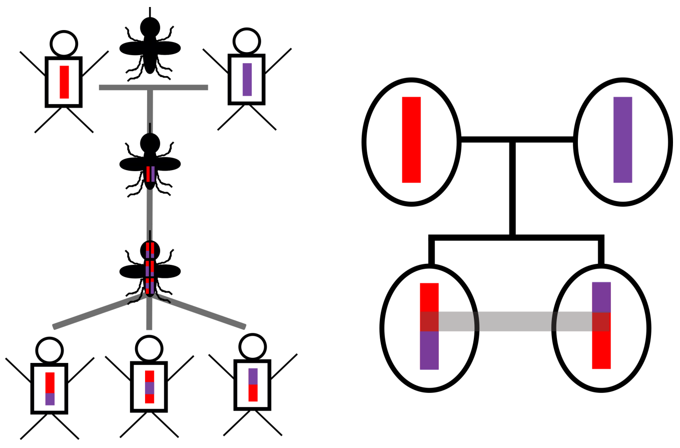
Click For Answer

Conceptual Question 2: *Do you recognize any pattern in heterozygote calls with respect to demes? How does this relate to the polyclonality lesson yesterday?*

Click For Answer

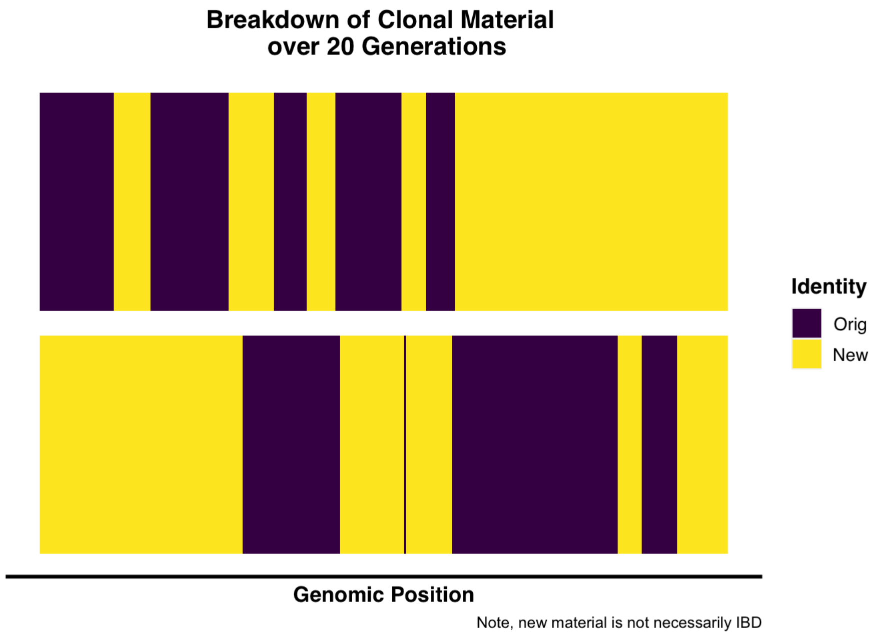
2 Intuition

Recombination produces a powerful signal for detecting genetic relatedness between individuals. In malaria, recombination occurs in the mosquito midgut. As a reminder, the schematic below shows a single mosquito that has bitten two different individuals infected with a single strain of the malaria parasite, resulting in a superinfection. The two strains (red vs purple) then undergo recombination in the mosquito midgut and produce progeny that are siblings and expected to share half of their genomes on average (*they are separated by 1 generation*). The pedigree on the right highlights two of the parasite siblings and has a grey bar indicating how much of their genome was inherited from the same parent in the same genomic location.



2.1 Visualizing Recombination

As a result, one way to conceptualize the genome is as blocks, or segments, that are joined together, where each block has its own unique ancestral history. Thus, recombination can be thought of as the "stick breaking" process that breaks the genome segments/IBD blocks up: as time increases, or the number of generations increases, the genomic blocks that are shared between individuals become smaller. For example, you can imagine that you have two sticks that are identical (*i.e.* twins, or clonal infections). In each new generation, you have to randomly break your stick at some point along its length. You then take your pieces of stick and flip a coin to determine if it will remain the same ("original") or inherit genetic material ("new"). Use the `view_recombo` (loaded from `R/utils.R`) function and explore the relationship between the number of generations, or time, and the length of recombination blocks that are IBD between the clones. The figure below shows the results for 20 generations.



For example, you could run the function with the following generational times

Hide

```
view_recombo(generations_apart = 0)
view_recombo(generations_apart = 1)
view_recombo(generations_apart = 2)
view_recombo(generations_apart = 3)
view_recombo(generations_apart = 5)
view_recombo(generations_apart = 10)
view_recombo(generations_apart = 20)
```

Conceptual Question 3: *How does the amount of "clonal" genetic material change, or amount of the genome that is identical, change between the paired samples as we increase the number of generations that separates them?*
Click For Answer

3 Calculations

3.0.1 Internal Data Manipulation

Here, we are going to convert our *vcf* object that we read in with the `vcfR` package to a *mip* object in order to later use the `MIPAnalyzer` Package (https://github.com/mrc-ide/MIPanalyzer/tree/master/R) and the maximum likelihood estimator of IBD from Verity et al. 2020 (https://pubmed.ncbi.nlm.nih.gov/32355199/). This is purely for convenience and is not changing the underlying data in the vcf.Please copy and paste the code below into your console.

Hide

```
mipvcf <- MIPanalyzer::vcf2mipanalyzer_biallelic(vcfR = vcf)
```

3.1 Identity by State

Identity by state (IBS) is the proportion of identical loci divided by all measured loci in the genome between two parasites: parasite_a and parasite_b, such that:

$$IBS = \frac{\text{Shared loci}_{ab}}{\text{Number of loci}}$$

Here, we will perform IBS calculations using our VCF as input.

Coding Question 3: Pick two samples from your VCF and write your own function to calculate IBS between the pair. I recommend starting with the `vcfR` object (vs. newly created `mipanalyzer_biallelic` object) and using the `extract.gt` function from the vcfR package to pull out the genotype calls (`element = "GT"`) as above. Then subset the data to two samples and compare loci.

Click For Answer

3.1.1 Run Package IBS

Next, we will use the IBS calculator from the `MIPAnalyzer` R package to calculate pairwise IBS for all combinations of samples in our simulated data. Please copy and paste the code below into your console for ease of running the algorithm.

Hide

```
# get IBS
ibs <- MIPanalyzer::get_IBS_distance(x = mipvcf,
                                     ignore_het = FALSE,
                                     report_progress = FALSE)
```

Tidy Results

The function returns the data in a "wide" format, or a distance matrix (https://en.wikipedia.org/wiki/Distance_matrix). While this a perfectly acceptable format, it is not considered "tidy".

Coding Question 4:: Wrangle the data into "long" format using `broom::tidy` function. Rename the columns for the long data to `c("p1", "p2", "ibsdist")` . Note, make sure to preserve the sample names from `colnames(vcf@gt)[2:ncol(vcf@gt)]` prior to tidying!

Click For Answer

It is always good practice to explore our results. Let's make a boxplot to explore the distribution of our calculated IBS results.
Coding Question 5:: Finished the code below to create a boxplot with IBS values on the y-axis.

```
ibs_long %>%
  ggplot() +
  geom_boxplot(***) +
  ylab("IBS Values") +
  theme_bw() +
  theme(axis.text.x = element_blank(),
        axis.ticks.x = element_blank())
```

Click For Answer

Conceptual Question 4: *Describe the distribution. Is this what you expected?*
Click For Answer

3.2 Identity by Descent

Identity by Descent (IBD) is more complicated to calculate than IBS and requires statistical modeling in order to account for recombination and population allele frequencies. The most common statistical model used is called a Hidden Markov Model (see the *Further Reading* section for recommended sources). Remember, a site can be IBS but not necessarily IBD as described in the IBS/IBD Overview section.
Here, we will use two different algorithms to calculate IBD: (1) an MLE inbreeding estimator; (2) a hidden Markov model.

MLE IBD

This first algorithm uses a maximum likelihood estimator of IBD (see Verity et al. 2020 (https://pubmed.ncbi.nlm.nih.gov/33057671/) for a mathematical description). The model assumes that loci are independent, which allows for IBD to be essentially calculated as the amount of inbreeding, or deviation from the expected allele frequencies, between allele frequencies per site. The model is a variation of the Generalized Hardy-Weinberg equation (https://en.wikipedia.org/wiki/Hardy%E2%80%93Weinberg_principle) for nonrandom mating (see Gillespie, Population Genetics: A Concise Guide (https://public.wsu.edu/~gomulki/mathgen/materials/gillespie_book.pdf) 1st edition, pg 86, for

further reading).
As above, we will use `MIPAnalyzer` R package to calculate pairwise IBD for all combinations of samples in our simulated data. For convenience, copy and paste the code below into your console to acquire these results.

Hide

```
#......................
# MLE IBD
#......................
ibd <- MIPanalyzer::inbreeding_mle(x = mipvcf,
                                   f = seq(0.01, 0.99, 0.01),
                                   ignore_het = FALSE,
                                   report_progress = FALSE)
```

Coding Question 6:: Let's now tidy/convert our data to "long" format again using the `broom::tidy` function. Make sure to set your diagonal to 1 (self comparisons), preserve the sample names, and name your columns `c("p1", "p2", "malecotf")`.

Click For Answer

After generating our new results, we always explore them!
Coding Question 7: Using the `geom_boxplot` code from above, create a boxplot with IBD values on the y-axis.

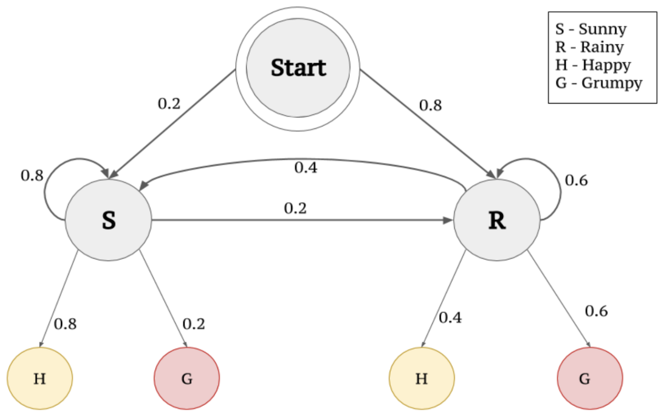
Click For Answer

Conceptual Question 5: *Describe the MLE distribution. Is this what you expected?*

Click For Answer

IBD from HMM

Next, we will use a hidden Markov model (HMM) from the R-package `hmmIBDR`, which is a wrapper for the original `hmmIBD` algorithm (for a full mathematical description of `hmmIBD` see Schaffner & Taylor 2018 (https://pubmed.ncbi.nlm.nih.gov/29764422/)) to calculate IBD estimates. As described in the lecture, HMMs can be thought of a statistical model that uses "observations" to infer the "hidden" state of some process, where the process is known or can be described. In our case, our observations are genotype calls between individuals (identical sites versus discordant sites) and our hidden state is whether or not that site is a region of common inheritance between individuals (IBD) or not. The underlying process is recombination, which we know breaks up genomic blocks, and can be described based on the known recombination rate in malaria. This site (https://towardsdatascience.com/markov-and-hidden-markov-model-3eec42298d75) provides a useful non-genomic example: imagine you call a friend every day that lives far away, and that friend has two "state's" of mind: grumpy or happy. You know that your friend's mood is dependent on the weather, but we don't know the weather in this far away place (it is "hidden" from us). However, we can infer what the weather was that day based on whether our friend was happy or grumpy when we spoke to them. Moreover, if we recorded our friend's mood for a week, we could use a model to infer the weather pattern over that entire week based on the observed sequence of our friend's "state" of mind as shown in the schematic below. For example, if our friend was G,G,G,H,H,H,H we may suspect that it rained for the first three days of the week but then there was a state change and it was sunny the last four days of the week.



Formatting for `hmmIBD`

Prior to running `hmmIBD` program (https://github.com/glipsnort/hmmIBD), we are going to have to do some data munging.
Coding Question 8, part 1: Make a within-sample alternative allele frequency (WSAF) matrix. First, start by extracting the alternative allele, or the number of alternative reads, at a particular locus per sample using the `vcfR::extract.gt(vcf, element = "AD")` function: (the numerator). Next, calculate the total allele depth per locus, per sample using the `vcfR::extract.gt(vcf, element = "DP")` function: (the denominator). Now, find the WSAF by dividing the numerator by the denominator and name this new matrix `wsaf` .

Click For Answer

Coding Question 9, part 2: In order to convert this dataframe into a " `hmmIBD` compliant dataframe, first, add the chromosome and position to your wsaf matrix (as the first and second columns respectively) and name it `gtmat` . Then tidy your"wide" matrix to "long" format excluding the chromosome and position (*i.e.* we want a dataframe with chromosome, position, sample, wsaf as our columns) with the function: `tidyr::pivot_longer(., cols = !c("chrom", "pos"))` . We are now going to coerce are WSAF calculations into homozygous genotype calls using rounding: `gthmm = round(value, 0)` . In addition, make sure to drop the `value` column (now redundant) and for `hmmIBD` liftover the chromosome names to integers: (`chrom = 1)` . In a turn of events, we will now make our data "wide" from long and write out this new table to your local drive for `hmmIBD` using the functions:
`tidyr::pivot_wider(data = ., names_from = "name", values_from = "gthmm")` and
`readr::write_tsv(x = gtmat, file = "data/gt_matrix_for_hmmIBD.txt", col_names = T)` , respectively.

Click For Answer

Coding Question 10, part 3: Finally, using your `wsaf` matrix, calculate the population level allele frequencies for each locus using the following function: `altafvec <- rowMeans(wsaf, na.rm = T)` . Remember, this is the alternative allele frequency, and because we are dealing with biallelic sites, we can calculate the referent allele frequency as 1-alt and place this all in a dataframe. The dataframe needs a chromosome column (as a numeric/integer [`chrom = 1`]), a column for locus' basepair position, a referent allele frequency column, and an alternative allele frequency column. Write this table out to your local drive with the following path: `data/af_matrix_for_hmmIBD.txt` .

Click For Answer

Explore the hmmIBD documentation (https://github.com/glipsnort/hmmIBD) for a justification of this munging that we did above and further clarification of the caveat below.

Caveat*: The `hmmIBD` algorithm assumes that all samples are monoclonal. As a result, it does not accept heterozygote calls natively. As a user, we have two options: (1) to set those heterozygote calls to missing (a conservative approach), or (2) make a strong assumption about what a heterozygote call means (an aggressive approach). Here, I have forced the samples to be monoclonal by estimating their within-sample allele frequencies and rounding to the nearest variant.*

Running hmmIBD

Now we will run the `hmmIBD` program to calculate pairwise IBD. For convenience, copy and paste the code below into your console to acquire these results.

Hide

```
#......................
# hmmIBD run
#......................
# NB, participants do not need to use the "quiet" function
tf <- tempfile(pattern = "output_simdat")
out <- quiet(hmmibdr::hmm_ibd(input_file = "data/gt_matrix_for_hmmIBD.txt",
                              allele_freqs =  "data/af_matrix_for_hmmIBD.txt",
                              rec_rate = 1e-2, # note the small recombo rate relative to what would be expect
        ed in malaria
                              output_file = tf))

# hmmIBD tidy
ibd_hmm_long <- tibble::tibble(
  p1 = out$fract$sample1,
  p2 = out$fract$sample2,
  hmm = out$fract$fract_sites_IBD)
```

Again, we always explore our results!
Coding Question 11: Use your `geom_boxplot` code to create a boxplot with IBD HMM values on the y-axis.

Click For Answer

Conceptual Question 6: *Describe the HMM distribution. Is this what you expected?*

Click For Answer

Contrast MLE vs HMM

As described above, the IBD calculations from the MLE model assume independent loci, while the HMM model leverages recombination to detect relatedness. However, `hmmIBD` requires monoclonal infections and we had to "coerce" our data to fit that assumption. Here, we will explore how the MLE IBD and the HMM IBD results differ.

Coding Question 12: Use the `geom_point` ggplot layer to create a plot that contrast the two different calculations. You will need to join the data by sample using the following function: `left_join(ibd_mle_long, ibd_hmm_long, by = c("p1", "p2"))` by finishing the code chunk below.

```
dplyr::left_join(***) %>%
  ggplot() +
  geom_point(***) +
  theme_bw() +
  labs(x = "IBD from HMM", y = "IBD from MLE")
```

Click For Answer

Conceptual Question 7: *Why would these two results differ? Are there any differing assumptions made in the models?*

Click For Answer

Contrast IBS vs HMM-IBD

Compare IBS and the `hmmIBD` results in a scatterplot.

Coding Question 13: Using the `geom_point` ggplot layer, create a plot that contrast the IBS estimations and the MLE IBD estimations. Add a regression line to explore the correlation using the `geom_smooth(aes(x = malecotf, y = ibsdist), method = "lm")` layer. Note, you will need to join the data by sample using the following function: `left_join(ibs_long, ibd_mle_long, by = c("p1", "p2"))` prior to using ggplot.

Click For Answer

Conceptual Question 8: *How do the estimates of IBS and IBD differ. What is the smallest value for IBS versus IBD? Why may these be different?*

Click For Answer

3.2.1 Summary/Check-In

At this point, you should have created three R result objects:

```
ibs_long
ibd_mle_long
ibd_hmm_long
```

If you are having computational trouble or time is a limiting factor, please see the `results/` directory, where you will be able to read in the results.

4 Applied IBD

In this section, we will focus on the utility of IBD for realistic applications related to control and elimination efforts. For this section, we will focus on the MLE-IBD estimates (versus HMM-IBD estimates).

4.1 IBD Distribution

First, let's look at the distribution of our IBD estimates. In particular, we will focus in on the "tail" of the distribution (*i.e.* high values). Plotting the IBD distributions help to convey how interbred our population is, and if we expect there to be highly related pairs that are of importance for revealing corridors of gene flow (assessed later).

Coding Question 14: Use the `geom_histogram` ggplot layer to create a plot that shows the distribution of MLE IBD estimations. We will add an inset plot to explore the tail of this distribution. Explore this blog (https://meghan.rbind.io/blog/cowplot/) to learn how to make plots with insets using the `cowplot` package.

Click For Answer

Conceptual Question 9: *How would you describe this distribution? Would you expect this many unrelated samples? What about the proportion of highly related pairs in the tail of the distribution?*

Click For Answer

4.2 Transmission Intensity

As we have seen from the recombination review at the beginning of the practical, as the number of recombination events increases, we expect for IBD due to decrease. As a result, in areas with higher transmission intensity, here modeled through differences in COI, we can expect for IBD to be less.

4.2.1 Within-Deme IBD vs COI

By grouping samples together by their home demes, we can estimate the amount of inbreeding within a deme, or the within-deme IBD. Here, we will read in the `data/metadata.RDS` file and calculate the mean IBD by deme to get the within-deme IBD. We will then plot the within-deme IBD versus simulated mean COI.

Coding Question 15, part 1: Read in the metadata using the following code: `readRDS("data/metadata.RDS")`. Then combine our metadata to the `ibd_mle_long` dataframe using two different `left_join` calls. Note, you will need to make two different "metadata" dataframe with a column named "p1" (sample 1) and one with a column named "p2" (sample 2). Rename the ".x" and ".y" extensions created by `left_join` to "_p1" and "_p2", respectively. After joining this data together, you should have a new dataframe with column names: `p1, p2, malectof, deme_p1, coimeans_p1, longnum_p1, latnum_p1, deme_p2, coimeans_pe2, longnum_p2, latnum_p2`. We are going to use this dataframe for the next few challenges, so please name it `ibd_mle_long_mtdt`.

Click For Answer

Conceptual Question 10: *Describe your new dataframe.*

Click For Answer

Code Question 16, part 2: Calculate within-deme IBD. Now, subset your dataframe to only contain rows where pairwise comparisons are from the same deme: `dplyr::filter(deme_p1 == deme_p2)`. Group by demes (`group_by(deme_p1)`) and use the `summarise` function to calculate the mean within-deme IBD and the mean COI (note, COI is the same within a deme). Then use the `geom_point` layer in ggplot to contrast COI and within-deme IBD calculations.

Click For Answer

Conceptual Question 11: *Is there a correlation here between mean COI and within deme IBD?*

Click For Answer

Conceptual Question 12: *Why did our expectation of the relationship of within-deme IBD and transmission intensity not match reality?*

Click For Answer

4.3 Isolation by Distance

Next, we will explore the concept of isolation by distance, which is the theory that as pairs move further away in space, they should be less related (*i.e.* closer pairs are more likely to reproduce). This concept is based on spatial relationships, but can also be conceptualized as time (pairs that are separated by multiple generations of time are less likely to be related).

Coding Question 17, part 1: Calculate greater circle distance (https://en.wikipedia.org/wiki/Great-circle_distance). First, we will read in our pre-calculated GC distances: `readRDS(data/deme_gc_dist.RDS)` . Use a `left_join` call to merge this data while making a new dataframe called `ibd_mle_long_mtdt_dist` . Now, we will "bin" our distances by converting this continuous value into a discrete form using the `cut` function:

```
cut(x = c(ibd_mle_long_mtdt_dist$distance),
breaks = c(0, 1e-26, seq(40, 120, by = 40), Inf),
right = F,
labels = c("Within", "40km", "80km", "120km", ">120km"))
```

Click For Answer

Coding Question 18, part 2: Using our newly created dataframe, `ibd_mle_long_mtdt_dist` , group by the `distance_cat` column created with the `cut` function above and calculate the following summaries (`summarise`): mean IBD, standard deviation IBD, standard error IBD, lower 95% CI for IBD, and upper 95% CI for IBD. You can review the power practical, part 1 for a refresher on these statistics.

Click For Answer

Coding Question 19, part 3: Using the `geom_pointrange` layer in ggplot, plot the mean IBD on the y-axis and the categorized GC distance on the x-axis. For the `geom_pointrange` aesthetics, you will need to specify: `y = meanIBD, ymin = L95CI, ymax = U95CI` .

Click For Answer

Conceptual Question 13: *How would you describe this relationship? Does it make sense?*

Click For Answer

4.3.1 Network Analysis

We can also use networks to determine the connectivity of our paired samples. Networks helps us visualize connections that we may not expect and are a useful tool for exploratory data analysis. Additionally, we can use community detection algorithms (https://towardsdatascience.com/community-detection-algorithms-9bd8951e7dae) to explore which samples may "cluster" together based on relatedness. One way to conceptualize community detection algorithms is to consider that networks measure "popularity" and that samples that are "cliques" should cluster together. Alternatively, a sample may be "friends" with everyone and becomes very central or important to the network. These distinctions may be helpful to determine which demes are particularly connected and may be contributing to sink-source dynamics (see Wesolowksi *et al.* 2018 (https://pubmed.ncbi.nlm.nih.gov/30333020/) for further information on sink-source dynamics). Read this STDHA blog (http://sthda.com/english/articles/33-social-network-analysis/136-network-analysis-and-manipulation-using-r) networks and on community detection algorithms and use the R packages: tidygraph (https://tidygraph.data-imaginist.com/index.html) and ggraph (https://ggraph.data-imaginist.com/) to try to generate the figure below.

Coding Question 20: Use the `tidygraph::as_tbl_graph` function to convert our `ibd_mle_long` into an a `tbl_graph` for easier network analyses. Then calculate the community membership using `tidygraph::group_louvain(weights = malecotf)))` function and plot the resulting network with `ggraph::ggraph(layout = 'kk')` . Make sure to color in the nodes by community using the `geom_node_point` layer and proper aesthetic: `ggraph::geom_node_point(aes(color = community)` .

Click For Answer

Coding Question 21: While that graph is interesting, it may be hard to interpret given all of the low-level relatedness connections. Let's prune those and keep only connections with IBD ≥ 0.1. Use the `dplyr::filter` function to perform this task.

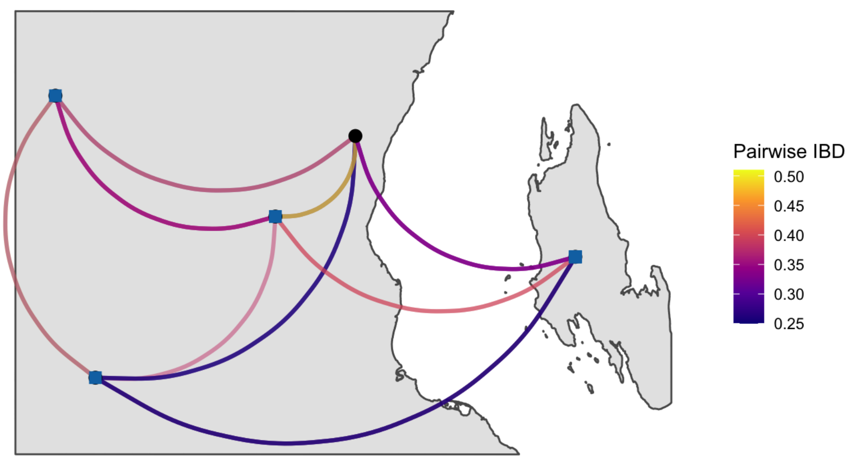
Click For Answer

Conceptual Question 14: *What do you notice about the above network? Are there any isolated cliques? Are there any very "popular" samples?*

Click For Answer

4.4 High Related Pairs

Here, we will subset to highly related pairs, defined as pairs that are at least meiotic siblings (*i.e.* share half of their genome). We will then map these between deme connections to determine if there is evidence of genetic sharing between demes. In this section, we will practice layering multiple dataframes onto a single plot to generate a figure like below.



Locations that contain highly related pairs within the deme are colored as blue squares; between deme connections are shown are curved lines

Coding Question 22, part 1: Here we will set up our data for making the map/plot. Read in the simulated map with the following function: `readRDS("data/sim_map_sf.RDS")`. Then make a condensed version of the metadata that just contains one row of coordinates for each deme. Next, make a dataframe containing highly related pairs between demes by subsetting the `ibd_mle_long_mtdt` to at least meiotic siblings: `dplyr::filter(malecotf >= 0.25)` and by excluding pairs that are from the same deme: `dplyr::filter(deme_p1 != deme_p2)`. Finally, create a dataframe of demes that contain highly related pairs by subsetting the `ibd_mle_long_mtdt` to at least meiotic siblings: `dplyr::filter(malecotf >= 0.25)` and by including only pairs that are from the same deme: `dplyr::filter(deme_p1 == deme_p2)`. You know have four objects ready for plotting: (1) map, (2) cluster locations, (3) highly related between deme pairs, (4) high related within deme pairs.

Click For Answer

Coding Question 23, part 2: We will now plot these dataframes. First make a map base using the `geom_sf` layer. Next add a layer for cluster locations using `geom_point(data = mtdtclst, aes(x = longnum, y = latnum)`. Now we will color in the demes that contain highly related pairs within: `geom_point(data = withinpairs, aes(x = longnum_p1, y = latnum_p1), color = "blue"`. Finally, we will add a connections between pairs across different demes from our highly related between pairs dataframe: `geom_curve(data = highbtwnpairs, aes(x = longnum_p1, y = latnum_p1, xend = longnum_p2, yend = latnum_p2, color = malecotf` Your plot contains both spatial and relatedness data in a easily communicated graph. Note, you can change the order of the layers in order to have either points or curves on top (remember, ggplot layers follow the order of the code).

Click For Answer

Conceptual Question 15: *What can highly related pairs tell you? How do they relate to transmission events? Can you use them to estimate genetic corridors or evidence of gene flow? Do you think there is active transmission between the mainland and island (see original study hypothesis)?*

Click For Answer

Addendum

Congratulations, you have reached the end of the practical. This section contains advanced material and additional resources for interested participants. If time permits, you can explore the advanced material challenge, the challenge questions, and/or the further readings.

Coding Challenges

In this section, you will have the opportunity to explore and practice the skills above.
Imagine that there are two populations: *A* and *B* with similar levels of falciparum incidence. However, there are only enough public health funds to intervene on one population. You have been asked by your National Malaria Control Programme (NMCP) to determine which population to intervene on. You first conduct a "travel survey" and determine that there are a high number of people that travel from *A* to *B*. As a result, you hypothesis that *A* is a "source" for *B*. Using the *VCF* below, create a series of figure that you would present to your NMCP to encourage them to interview on population *A* instead of *B*. Make sure to calculate IBD with multiple approaches to confirm your results.

```
# participants DO NOT need to include the quiet bit
ssvcf <- quiet(vcfR::read.vcfR("data/simulated_sink_source_ibd.vcf.gz"))
```

When finished with your figures, please ask an instructor for feedback.

IBD & Polyclonality

As was described in the case of `hmmIBD`, polyclonality greatly complicates IBD estimates and is an active area of research. Recent solutions include considering the malaria genome as diploid, which allows for heterozygous alleles: `isoRelate` (https://github.com/bahlolab/isoRelate) or phasing polyclonal infections and estimating IBD from the resultant haplotypes: deploidIBD (https://github.com/DEploid-dev/DEploid). This is an active area of research and remains a challenge within the malaria genomic epidemiology field.
Below is a table of pros-cons-details of various IBD programs specified for malaria that have been peer-reviewed as of August 5, 2022.

Program	Pros	Cons	Details
Inbreeding estimation (https://pubmed.ncbi.nlm.nih.gov/32355199/)	+ Fast	+ Assumes independent loci	Model is not considering recombination blocks (backbone of IBD)
hmmIBD (https://github.com/glipsnort/hmmIBD)	+ Fast + Easy to use + Statistically robust + Canonical program for malaria ibd	+ Specified for monoclonals	Users can use in polyclonal samples, but is misspecified under model and results are typically conservative estimates of IBD
isoRelate (https://github.com/bahlolab/isoRelate)	+ Statistically robust + Handles polyclonals + Statistical testing for "important" loci	+ Can become computationally intensive	Model assumes a "diploid" state for COI > 1, and thus makes a simplifying assumption about polyclonality
DeploidIBD (https://github.com/DEploid-dev/DEploid)	+ Statistically robust + Handles polyclonals + Phased sample results + Within-sample estimates of IBD	+ Computationally intensive	Model phases polyclonal samples into monoclonal samples (*i.e.* a sample with COI = 2 becomes two new phased strains: smpl1.1, smpl1.2). As part of phasing, program provides within-sample IBD. From phased sample results, users can use any existing IBD program (`hmmIBD` , `refinedIBD` , etc). Program's primary goal is phasing with within-IBD estimates provided as a "bonus".

Further Reading

HMMs

Hidden Markov models (HMMs) are a common statistical approach in population genetics, and for detecting IBD. For further reading, see Rabiner 1989 (https://web.mit.edu/6.435/www/Rabiner89.pdf).

Coalescent Theory

Coalescent theory is one of the central pillars of population genetics and provides a framework for how loci (genes, individuals, etc.) have been derived from a common ancestor backwards in time, classically using the assumptions of the Wright-Fisher model.
The following are an excellent launching point for further reading:
Wakeley: Coalescent Theory: An Introduction (https://www.amazon.com/Coalescent-Theory-Introduction-John-Wakeley/dp/0974707759)
Li & Stephens: 2003 PSMC (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1462870/)
Li & Durbin: 2011 PSMC (https://pubmed.ncbi.nlm.nih.gov/21753753/)